Using IDA for Performance Improvement in Multimedia Servers

Antonio Puliafito, Salvatore Riccobene, Lorenzo Vita
Istituto di Informatica e Telecomunicazioni
Facoltá di Ingegneria - Universitá di Catania
V.le A. Doria 6, I-95125 Catania - ITALY

Miklós Telek
Department of Telecommunications
Techical University of Budapest
1521 Budapest - HUNGARY

January 31, 1996

Abstract

In order to meet the new requirements of multimedia applications, high-performance storage devices are needed with a very high cost per storage unit. However, if the access flow has particular requirements it is possible to adopt specific data storage and management algorithms which can optimize the performance of traditional systems, thus allowing them to be used in multimedia servers.

The paper proposes, and evaluates by a simulation model, the use of the *Information Dispersal Algorithm* (IDA) for the management of an I/O subsystem based on disk array technology. The architecture is specifically designed for the management of multimedia data with the characteristics typical of Continuous Media (CM) data services.

1 Introduction

Multimedia systems derive from the need to adapt the presentation of data to the real nature of the human user, through combined flows of audio and video data. Examples of this kind are *Continuous Media* (CM) which are arousing great interest at present. They feature a continuous flow of data from a service manager (mass storage system) towards a multimedia user interface [1]. *CM* systems require the transfer of large amounts of data within strict time constraints (*firm deadlines*): the data transmitted has to be available in a certain time window. Thus, in such systems meeting deadlines assumes a role equally as important, if not more important, than the correctness of data.

Data retrieval from storage servers represents one of the main causes of deadlines violation. These devices, essentially based on mechanical equipments (disks, tapes, etc,), are in fact so limited in their technological evolution as to cause bottlenecks for the performance of many systems, even ones which do not provide multimedia services. A possible solution to the problem of mass memory can be found by parallelizing I/O subsystems, using *Disk Array* techniques [2]. Disk arrays are sets of hard disks arranged in such a way that the various I/O operations can be performed either by different units (parallelization of requests) or by several units at the same time (parallelization of services).

As compared with traditional applications, the storage of data for *CM* systems shows particular features which are essentially linked to the following factors:

- the transfer rate required is very high;
- the flow of information is sufficiently continuos in time;
- user requests to access the server are almost exclusively read-type.

An example of a *CM* application is *Video on Demand* (VOD) [3], which allows the user to access the movie he/she likes at the more convenient time, with the possibility to customize the flow of information at own wishes. *Play, Pause, Forward, Rewind* functions are in fact implemented in such a way that the user can completely control the movie execution as if it were processed by a VCR. Thus, the problem consists in defining a reference architecture which can exploit these peculiar features of *VOD* systems and speed up the access to the storage devices.

The aim of this paper is to define and assess by simulation a disk array system for the memorization of specific data for *CM* applications. Focusing on *VOD* service systems, a redundancy scheme based on an *Information Dispersal Algorithm* (IDA) [8] is used, its main task being to enhance the performance of the system. An index of the Quality of Service (QoS) offered by the system is also defined and evaluated.

The paper is organized as follows: Section 2 introduces the problem of data storage in *CM* systems for *VOD* applications and discusses about the IDA coding technique. Section 3 describes the Disk Array system proposed and summarizes the basic features of the *IDA* encoding scheme adopted. Sections 4 and 5 present a simulative analysis of the system. The authors' conclusions and suggestions for further research are given in Section 6.

2 Disk Arrays for CM systems and the IDA coding scheme

Figure 1 depitches the three-level reference architecture that will be assumed in the paper.

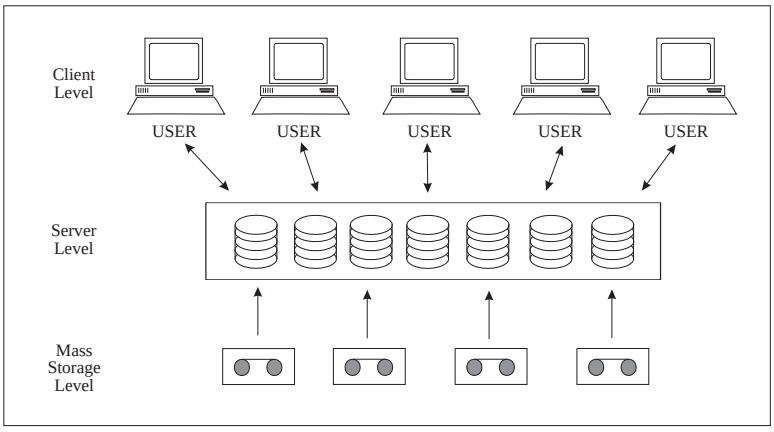

Figure 1: Hierarchical organization of a *VOD* system.

The lowest level comprises mass storage devices which feature a high capacity, a low access speed and, above all, a low cost/storage ratio. The next level is the real server of the system and its devices not only have a high storage capacity but also offer considerable speed in

access to data and services. The third level refers to the final user or group of users with particular features in common, and provides an access interface and buffer memories which are extremely rapid but have a limited capacity, for the temporary storage of data.

In the following, we will focus on the middle level of this architecture, which will be based on the use of disk array technology. The disk array system described is a set of interconnected disks with an architecture similar to the partial dynamic declustering presented in [9]. Assuming the partial declustering as the starting point, this technique allows a dynamic management of the formation of groups, varying not only the disks in the groups, but also their number.

We assume that data relating to "normal" traffic is stored according to traditional technique (parity encoding), while multimedia data is recorded using an information dispersal technique based on the *Information Dispersal Algorithm* (IDA) described in [8].

IDA is based on particular coding of information which, in the recording phase, provides the opportunity to vary arbitrarily the number of units used to achieve data redundancy. The system administrator can decide the number of additional units file by file, according to the degree of security required. Coding techniques are widely used in disk array systems in order to enhance reliability. In practice, thanks to the introduction of a certain amount of redundancy in the writing phase, it is possible to reconstruct the contents of a disk perfectly on the occurrence of a fault.

However, IDA enjoys a second property which consists of the possibility of reconstructing original data, in a perfectly equivalent way from the performance viewpoint, on the basis of any minimal subset (K) of disks. This feature makes IDA very attractive for performance purposes: the opportunity of run-time choice of the optimal subset K^* that will reduce the system's response times represents in fact a great advantage in terms of performance. In this paper we would like to focus on this point, trying to investigate how to stress this feature in order to reduce the service time offered by the system.

Similarly to other coding techniques, we are perfectly conscious that the use of the *IDA* technique to disk arrays is not particularly efficient in the data writing phase, as each modification of data involves all the storage units. For *OLTP* applications, in which a number of small requests are served, the penalization introduced may be excessive.

IDA coding is therefore inadequate for general-purpose use. For the *VOD* multimedia applications, in which we are interested here, however, disk arrays are generally loaded off-line with operations that require the transfer of large amounts of data, so the slowing down introduced by *IDA* is not significantly penalizing. On the basis of these considerations it is obviously convenient to limit the use of *IDA* coding to multimedia traffic alone, using classical (parity encoding) techniques for the rest of the data. In connection with this it should be pointed out that *Partial Dynamic Declustering* [9] lends itself particularly well to the simultaneous support of different data allocation techniques, thus combining well with *IDA*. The only necessary condition is recognition of the type of request (multimedia or otherwise).

3 System modeling

In managing a system based on *IDA* three different strategies are possible:

1. The first and most simple one consists of sending a read request to all the disks involved. Information can be decoded as soon as any M disks have processed the subrequest. In such a situation, the subrequests processed by the remaining N-M disks constitute an

overhead for the system, which tends to become excessive as the workload increases, thus annulling the advantages *IDA* provides.

The possible alternatives consist of:

2. deciding on the basis of certain status parameters which disks to use for the service before the request is sent;

3. removing later requests from the queue, (obviously after the minimum number of responses needed for decoding is reached).

Solution 2 certainly avoids overloading the system. On the basis of various parameters such as the size of the queue for each disk, the manager decides which disk to address the requests to. The number of subrequests sent to the system is the necessary minimum. However, the choice of disks is made a priori and may not be the best choice.

Solution 3 consists of sending the read request to all the disks and waiting, as seen above, for the first M to respond, automatically guaranteeing the optimal choice for serving the request. Then, once the minimum number of blocks needed to reconstruct the information has been received, all the other requests are eliminated from the remaining N-M disks so as to avoid overloading the system by processing data that is no longer necessary.

Unfortunately this second phase is not always feasible. There is a possibility that the request has already been removed from the queue and is being served. Although in theory it can still be blocked, there will still be a certain amount of time during which the disk will have worked unnecessarily (overhead), blocking other requests in the queue. This time interval ranges from a minimum, corresponding to the seek time, to a maximum which is equal to the time required to transfer the request to be eliminated. In addition, even simple elimination operations require a finite length of time, the value of which may not be negligible.

According to the previous discussion, the system taken in exam has several parameters which can be tuned, such as the total number of disks, the minimal subset of disks that are required to answer, the retrieve strategy adopted etc. In the following we analyze the results obtained by simulating various system organizations. In particular, besides the various architectural possibilities outlined above, we try to assess the impact of an increase in redundancy on the cost/performance ratio.

To evaluate the system a special event-driven disk array simulator was devised. As regards the construction and functional features of the disks, we based our model on the IBM 0661 3.5" 320MB SCSI Hard Disk model. Table 1 summarizes the main hardware features.

The seek time (T_s) for each disk is computed using the following formula:

$$T_s(x) = \begin{cases} 0 & \text{if } x=0 \\ a\sqrt{x-1} + b(x-1) + c & \text{if } x>0 \end{cases}$$

where x is the seek distance in cylinders and a, b and c are chosen to satisfy the single-cylinder-seek-time, average-seek-time and max-stroke-seek-time constraints according to the explanations given in [11]. Various tests were run varying the workload, the number of disks used and the degree of redundancy.

The system is able to process two kinds of requests: multimedia and data requests. The total workload to the system varies between 6 and 120 Mb/sec. Multimedia workload, which ranges from a minimum of 30% to a maximum of 70% of the total, assumes a periodical arrivals for each CM session. The average rate is assumed to be equal to 1.5 Mb/sec for each

Bytes per sector	512
Sectors per track	48
Tracks per cylinder	14
Cylinders per disk	949
Disk capacity	320 MB
Revolution Time	13.9 ms
Single cylinder seek time	2.0 ms
Average seek time	12.6 ms
Max stroke seek time	25.0 ms
Substained transfer rate	1.8 MB/s

Table 1: Disk parameters

Total number of disks in the system	30
Redundancy introduced	33.3 %
Group size (with redundancy)	8
Size of the minimal sub	6 (see section on *IDA*)
Minimum size of requests	10 blocks
Maximum size of requests	20 blocks
Blocks size	3 kByte

Table 2: Simulation parameters

session. A round-robin service policy is implemented which processes CM requests according to their deadlines constrains, eventually preeempting data requests under service [6]. For the data traffic an exponentially distributed arrival rate is assumed.

4 Simulations

A first series of simulations was run to assess the effect of workload on response times. The parameters assumed are described in Table 2.

With these parameters two series of tests were run to compare the two possible management strategies in *IDA*, i.e. sending subrequests to all the disks or only to the less loaded ones (optimal minimal subset). Below *MAX* will indicate the former case and *MIN* the latter.

Figure 2 shows the different behaviour of the two solutions, comparing the average response time for varying workloads when the MAX and MIN management strategies are assumed. With low workloads the *MAX* solution behaves considerably better, although it tends to reach saturation more rapidly than the *MIN* solution. This may be accounted for the low overhead introduced by sending subrequests to all the disks. On account of the correlation between the disk queues, the subrequests referring to a single operation reach the service phase almost simultaneously (after the queueing phase) so the overhead reduction mechanism described in Section 3 is not activated. In fact, once a particular value of workload is reached (70 Req/sec in the scenario taken in exam), the MIN solution performs alway better than the MAX one. In this case in fact, it is very likely that the subrequests from a single request will occupy

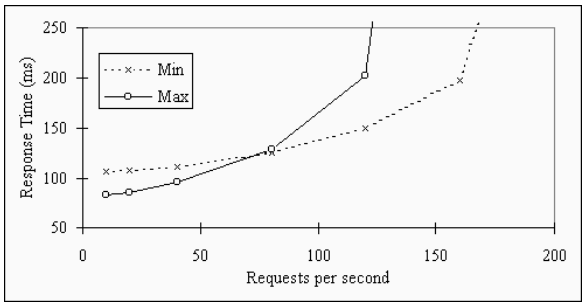

Figure 2: Response time vs. Workload

highly variable positions in the disks queues, due to the high number of requests arriving to the system. Removing subrequests from the queues once the minimum number of requests has been processed, will reduce the total overhead.

Although the results showed in Figure 2 illustrate important features of the way the system behaves, they are not able to provide a complete characterization of the system: no informations can be obtained on the system capability to meet the given deadlines. In other words, the time by which the system responds on average is not so useful as knowing the time t^* by which the system responds with a probability of P^*, having fixed P^* as equal to 95% - 99%. This probability P^* is directly linked to the Quality of Service (QoS) factor which identifies the system. The higher the QoS one wants to provide the user with, the higher P^* must be. It is therefore very useful to evaluate the response time probability distribution (see Fig. 3).

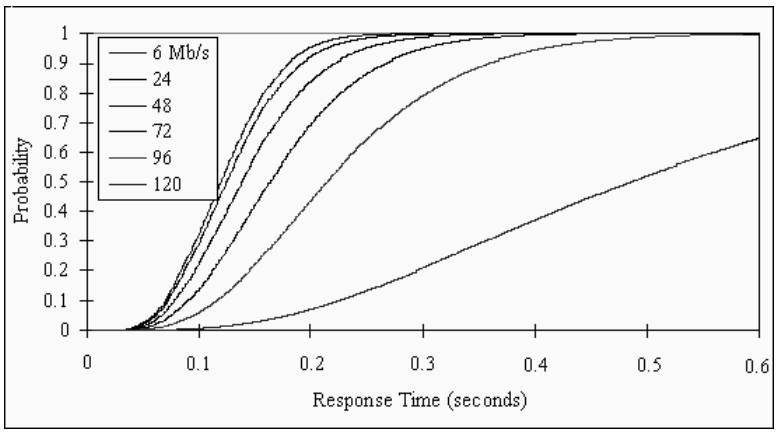

Figure 3: Response Time Distributions

Figure 3 shows the response time distribution function for various workloads (ranging from 6 to 120 Mb/s). The parameters of the system are the same as those used in the previous simulation experiment. The architectural solution is the *MIN* one. As can be observed, the higher is the workload the lower is the probability to meet a given deadline. Figure 3 makes it clearer why it is advisable to refer to high service probability values rather than to average ones. In other words, this graph provides insight on the way in which the quality of service degrades and, conversely, on the maximum workload the system is able to support. At a workload of 72 Mb/s, for example, even though the response time (T_r) is on average low, if

high probabilities are fixed, the T_r grows considerably with the risk of violating the deadline for some requests.

For each simulation experiment the confidence intervals were evaluated. However, the intervals are not shown in the plots because of their reduced value. For example, for the simulation experiment with workload equal to 72 Mb/s in Fig. 3, the larger interval with a 99% confidence interval was obtained for T_r=154 ms and was equal to (0.411, 0.414).

5 Sizing the System

In the previous Section was shown how the presence of a certain degree of redundancy in the storage phase, determines huge improvements from a performance point of view. However, it is not possible to use a eccessive degree of redundancy, because of the prohibitive amount of storage space dedicated to the coding informations. An exhaustive evaluation requires an insight analysis of the cost-performance ratio. In this Section we propose a figure of merit, strictly related to the Quality of Service the system provides, which considers the joint effect of the main parameters of the system. Specifically, what we want to deal with are:

- response time T_r;
- workload WL;
- total number of disks in system N_D;
- redundancy used R.

We indicate with Q the QoS indicator that will be evaluated according to the following expression:

$$Q = \frac{WL}{T_r \cdot N_D \cdot R}$$

Q depends linearly on the workload (WL), meaning that the greater is the workload served by the system, the more efficient it is. Similarly, Q is inversely related to T_r, N_D, R because an increase of these parameters indicates a non-optimal sizing of the system.

Different architectural solutions were hypotized, varying the degree of redundancy assumed. A given reference scenario is indicated by the pair i, j, where i represents the total number of disks storing a given information and j is the size of the minimal subset required by the IDA coding technique. For each reference architecture, Table 3 shows the redundancy introduced and the real amount of storage space that the system provides, assuming a total number of disks equal to 30 units. A workload of 60 Mb/sec is considered.

Fig. 4 is a diagram of the values of Q and the response time T_r when a service probability of 99% is assumed. The reason to plot Q and T_r in the same figure is to outline the limited information provided by only considering pure performance indices. The abscissa shows the different architectural solutions taken in exam.

As can be observed, albeit the T_r curve is continuosly decreasing when the degree of redundancy increases, the quality of service Q shows a maximum for the architectural solution $8, 6$, which provides a redundancy equal to 1.33. This organization provides the best cost/performance ratio.

Label	Redundancy	Available Space
6,6	1	30
7,6	1.16	25.71
8,6	1.33	22.50
10,6	1.66	18
12,6	2	15

Table 3: System parameters

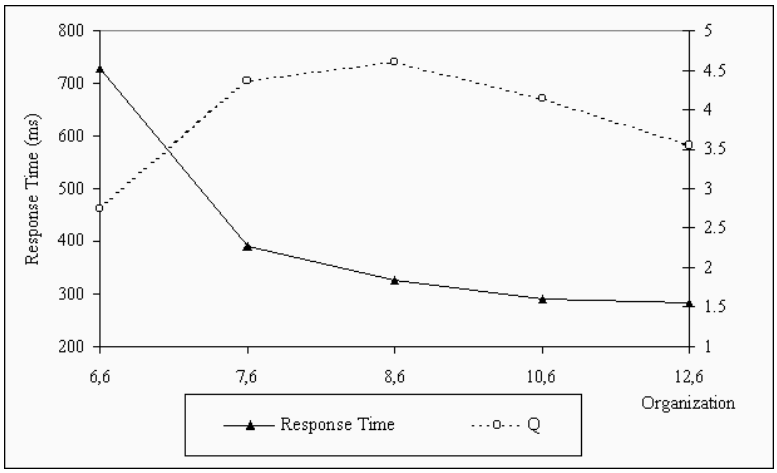

Figure 4: Quality of the different disks array organizations

6 Conclusions

The system proposed in this paper makes it possible to implement servers for multimedia systems. The combined adoption of partial dynamic declustering and *IDA* coding, made possible by the inherent features of *CM* data, gives great advantages in terms of response time reduction. Further improvements can be obtained by managing the service queues with pre-fetching algorithms since, on the basis the requests currently being served it is possible to foresee future requests with a certain degree of accuracy. All this allows high-quality multimedia services to be provided.

References

[1] D.J. Gemmel, H.M. Vin, D.D. Kandlur, PV. Rangan, L.A. Rowe, "Multimedia storage servers: a tutorial", *Computer*, Vol.28 No.5, May 1995.

[2] P.M. Chen, E.K. Lee, G.A. Gibson, R.H. Katz, D.A. Patterson, "RAID: high performance, reilable secondary storage", *ACM Computing Surveys*, Vol.26 No.2, June 1994.

[3] T.D.C. Little, D. Venkatesh, "Prospects for interactive Video-on-Demand", *IEEE Multimedia*, Vol.1 No.3, Fall 1994.

[4] A. Vogel, B. Kerhervé, G. von Bochmann, J. Gecsei, "Distributed Multimedia and QoS: a Survey", *IEEE Multimedia*, Vol.2 No.2, Summer 1995.

[5] *Computer Communications, Special Issue: Multimedia Storage Servers*, Vol.18 No.3, Mar 1995.

[6] K.K. Ramakrishnan, L. Vaitzblit, C. Gray, U. Vahalia, D. Ting, P. Tzelnic, S. Glaser, W. Duso, "Operating system support for a video-on-demand file service", *Multimedia System*, Mar 1995, Pg. 53-65.

[7] A.L. Narasimha Reddy, J.C. Wyllie, "I/O issue in a multimedia system", *Computer*, Vol.27 No.3, Mar 1994.

[8] M.O. Rabin, "Efficient dispersal of information for security, load balancing and fault tolerance" *ACM Journal of the Association for Computing Machinery*, Vol.36 No.2, June 1994.

[9] V. Catania, A. Puliafito, S. Riccobene, L. Vita, "Design and performance analysis of a Disk Array system", *IEEE Transaction on Computer*, Vol.44 No. 10, Oct. 1995.

[10] D. Stodolsky, M. Holland, W.V. Courtright, G.A. Gibson, "Parity logging DisK Array", *ACM Transaction on Computer System*, Vol.12 No.3, Aug.1994

[11] E.K. Lee, R. Katz, "An Analitycal performance model of Disk Array", *Proc. ACM Sigmetrics Conf. on Measurement and modeling of computer systems*, May 1993.

[12] D. LeGall, "MPEG: a video compression standard for multimedia applications", *Communication of the ACM*, Vol.34 No.4 Apr. 1991.

[13] R. Aravind, G.L. Cash, D.L. DuttWeiler, H.M. Hang, B.G. Haskeil, A. Puri, "Image and video coding standards", *AT&T Technical Journal*, Jan-Feb 1993.